REMARKS

Claims 1-10, 12-29 and 33-34 are pending in the present application, with claims 1, 12, 22, 26, and 33 being the independent claims. Claims 10 and 12-29 have been amended, claims 30-32 have been canceled, and claims 33-34 have been added. No new matter has been entered.

In the Office Action, dated July 20, 2006, claims 30-32 stand rejected under 35 U.S.C. § 101, claims 10 and 12-32 stand rejected under 35 U.S.C. § 112, first paragraph, as allegedly introducing new matter, and claims 1-10 and 12-32 stand rejected under 35 U.S.C. § 102(e) as allegedly anticipated by U.S. Patent No. 6,308,274 (Swift). These rejections are respectfully traversed.

Rejection of Claims 30-32 under 35 U.S.C. § 101

Claims 30-32 stand rejected under 35 U.S.C. § 101 as allegedly being directed to non-statutory subject matter. Claims 30-32 have been canceled, thereby obviating this rejection. New claims 33-34 are "computer readable medium" claims that are believed to comply with the requirements of 35 U.S.C. § 101. Reconsideration and withdrawal of the rejection based on 35 U.S.C. § 101 is thus respectfully requested.

Rejection of Claims 10 and 12-32 under 35 U.S.C. § 112, First Paragraph

Claims 10 and 12-32 stand rejected under 35 U.S.C. § 112, first paragraph, as allegedly failing to comply with the written description requirement for use of the term "tangible." The term "tangible" has been deleted from claims 10 and 12-29 and claims 30-32

have been canceled, thereby obviating this rejection. Reconsideration and withdrawal of the rejection based on 35 U.S.C. § 112, first paragraph, is thus respectfully requested.

Rejection of Claims 1-10 and 12-32 based on Swift

Claims 1-10 and 12-32 stand rejected under 35 U.S.C. § 102(e) as allegedly being anticipated by Swift (USP 6,308,274). Claims 30-32 have been canceled, thereby obviating this rejection with respect to those claims. The remainder of the claims (1-10, 12-29 and 33-34) are believed to be allowable for the reasons given below.

As previously noted, Swift describes that prior art access systems based on access tokens did not have a way of reducing privilege for a user where increased privilege is unnecessary. To solve that problem, Swift provides a mechanism to enforce "least privilege," or in some way reduced access, via restricted access tokens. Restricted tokens enable a security mechanism to determine whether a process has access to a resource based on a modified, restricted version of an existing access token. The restricted token is based on an existing token, and has less access than the existing token.

The restricted token is created by changing an attribute of one or more security identifiers that allow access in the parent token to a setting that denies access in the restricted token. Similarly, the restricted token can be created by removing a privilege from the restricted token that is present in the parent token. In addition, restricted security identifiers may be placed in a restricted token.

While the restricted token based system of Swift is made more versatile by allowing the creation of restricted access tokens, Swift remains an example of a system that enforces static access policy in that every time a token, e.g., a restricted token, of Swift is evaluated to

determine whether a particular task can be performed by a user, or whether an application may access a particular object, *the token or restricted token is evaluated using static data with static access policies*. In particular, Swift enforces a static access policy, as described in the background section of Applicants' specification (See pages 1-2, first three paragraphs of background).

The Examiner responded in the Official Action that Swift evaluates a restricted token differently "when there is a match for a security IDs (*sic*) in an access control entry (ACE) depending on whether the ACE is an allow ACE or a deny ACE" and thus implements a dynamic policy with dynamic data as claimed. Applicant respectfully disagrees. Applicant submits that Swift does not allow one to grant access privileges based on dynamic data (*e.g.* time) or dynamic policies (*e.g.*, restricted time access). The restricted tokens taught by Swift do not allow for such variability. Swift says nothing of dynamically varying access policies for a particular application.

The Examiner also responded in the Official Action that "a restricted token is dynamic data because different restricted tokens will be generated for the same user according to dynamic factors such as different types of operations and/or applications." Applicant submits that the Examiner has misunderstood the teachings of Swift. The restricted token does not change based on dynamic data. On the contrary, different restricted tokens are associated with *different* processes and do not provide for dynamic access policies regarding a particular application as claimed. The restricted tokens change only if the requesting process changes (and the new requesting process has a different associated access token) – the restricted tokens themselves are NOT "generated for the same user according to dynamic factors." On the contrary, the restricted tokens are preset and associated with

processes, not users. Restricted tokens as taught by Swift do not vary a client authorization context based on dynamic policies and/or dynamic data for a particular application as claimed.

To clarify the distinction between static data as used by Swift and dynamic data as claimed, the Examiner is asked to consider the following example. Swift creates a restricted token that is associated with a process, and when the process with the restricted token attempts to perform an action on a resource, a security mechanism compares the restricted token information with security information associated with the resource to grant or deny access. This process is no different than the process used for a process associated with the token from which the restricted token is based. No adjustment for dynamic factors such as client attributes or system environment variables is provided. Also, no mechanism is provided for changing the way in which a series of policy assertions are evaluated. Swift also does not provide a mechanism whereby a requestor must match all rather than any of the identities in order to grant access to a resource or to make a policy assertion (such as limiting access time) based on dynamic data (such as the time of day) for policy evaluation. In particular, Swift does not account for changes in the access data or policies from one access to the next for a particular application/process. On the other hand, the claimed method and apparatus provide for updating the client authorization context and/or access request based on dynamic data and dynamic policies for a particular application that may change from one access evaluation to the next. Thus, contrary to the Examiner's assertions, Swift does not evaluate a client authorization context or access request based on dynamic data and dynamic policy tailored to an application as claimed.

This distinction between static access policy and data and dynamic access policy and data is found in each of the independent claims, as emphasized below in bold:

1. A method for dynamically managing access to a resource in a computer system, the system having a client thereof making an access request for the resource, the method comprising:

determining, via an application programming interface, based upon dynamic data and first dynamic policy whether a client authorization context is to be updated, wherein said first dynamic policy is tailored to an application through which the resource is accessed;

identifying an access control entry as a callback access control entry; and

in response to identifying the access control entry as a callback access control entry, evaluating, via said application programming interface, **based upon dynamic data and second dynamic policy whether said callback access control entry bears on said access request, wherein said second dynamic policy is tailored to said application**.

12. A computer readable medium having computer executable instructions stored thereon that when executed by a computer cause the computer to carry out a method for dynamically updating a client authorization context in a computer system, the method comprising:

computing a client authorization context after the request for the resource is received from the client;

determining, via an application programming interface, based upon dynamic data and dynamic policy whether said client authorization context is to be updated, wherein said dynamic policy is tailored to an application through which the resource is accessed; and

updating said client authorization context according to said determination.

22. A computer readable medium having computer executable instructions stored thereon that when executed by a computer cause the computer to perform a method of dynamically managing access to a resource in a computer system, the system having a client thereof making an access request for the resource, the method implemented by the computer comprising:

comparing the authorization context of the client to at least one access control entry of an access control list;

identifying an access control entry as a callback access control entry; and

in response to identifying the access control entry as a callback access control entry, determining, via an application programming interface, **based upon dynamic data and dynamic policy whether said callback access control entry bears on said access request, wherein said dynamic policy is tailored to said application**.

26. For an application in a computer system having a resource manager that manages and controls access to a resource, a computer readable medium having computer executable instructions stored thereon that when executed by a computer causes the computer to carry out a dynamic authorization callback mechanism that provides extensible support for application-defined business rules via a set of APIs and DACLs including a **dynamic groups element, which enables an application to assign temporary group membership, based on dynamic factors, to a client for the purpose of checking access rights**.

33. A computer readable medium having computer executable instructions stored thereon that when executed by a computer causes the computer to provide **dynamic authorization of an application in a computer system based upon application-specific or business rules that incorporate dynamic data , the dynamic data including an identifier for identifying whether a dynamic access check callback function should be invoked for conducting said dynamic authorization of said application, data from client operation parameters, authorization policy data stored in a callback a callback access control entry, and any other authorization policy data managed, computed or retrieved by the application**, the computer executing said computer executable instructions to perform the steps of:

the application using an initialization routine to register with a resource manager **dynamic groups that enable the application to assign temporary group membership based upon transient or changing factors to a client for the purpose of checking access rights and to register with said resource manager dynamic access check callback functions that enable the application to perform customized procedures for checking access rights based on said transient or changing factors**;

adding said dynamic access check callback functions to the resource manager's registered callback list; and

automatically invoking a dynamic access check callback function by access check application programming interfaces that initialize a client authorization context from a system level authorization context or a user's security identifier, **whereby when a user attempts to connect to the application, the registered dynamic access check callback function is invoked such that the client context is augmented with client contextual data dynamically computed using said dynamic data**.

Since the system of Swift is based on static policy and data, as described in detail above, Swift cannot be said to teach or suggest at least the above-bolded elements of claims 1, 12, 22, 26, and 33. Claims 2-10, 13-21, 23-25, 27-29 and 34 depend from claims 1, 12, 22, 26, and 33, respectively, and are believed allowable for the same reasons. Reconsideration and withdrawal of the §102(e) rejection based on Swift is respectfully requested.

Upon reconsideration of the claims in view of Swift, the Examiner is asked to further note that the Swift patent is commonly owned prior art that falls within the provisions of 35 U.S.C. §103(c).

CONCLUSION

Applicants believe that the present Amendment is responsive to each of the points raised by the Examiner in the Official action, and submits that Claims 1-10, 12-29, and 33-34 of the application are in condition for allowance. Favorable consideration and issuance of a Notice of Allowability are earnestly solicited.

Date: December 20, 2006

/Michael P. Dunnam/
Michael P. Dunnam
Registration No. 32,611

Woodcock Washburn LLP
One Liberty Place - 46th Floor
Philadelphia PA 19103
Telephone: (215) 568-3100
Facsimile: (215) 568-3439